FedTrace: Generated-Content-based Watermark Verification for Traitor Tracing in Federated Learning

Tianzhe Xiao[1] **Haozhao Wang**[1] **Yichen Li**[1] **Qiyu Qin**[1] **Debin Liu**[2]* **Ruixuan Li**[1]

[1]School of Computer Science and Technology,
Huazhong University of Science and Technology, Wuhan, China
[2]School of Computer Science and Artificial Intelligence,
Zhengzhou University, Zhengzhou, China

{d202381469,hz_wang,ycli0204,qiyu_qin,rxli}@hust.edu.cn
debinliuhust@gmail.com

Abstract

As large generative models become widely deployed and customized, federated learning is increasingly used to adapt them while keeping user data local. Because clients repeatedly receive up-to-date adapters, a malicious client can copy a dispatched adapter, use it offline, and monetize generated images without exposing the stolen weights or a queryable service. The risk is amplified when multiple clients collude by merging or averaging leaked adapters, which can blur their identities. Existing federated watermarking and traitor-tracing methods usually assume white-box access to suspect weights or black-box query access to the deployed model. In realistic generative-model theft, however, the defender may only observe images that have already circulated. We propose *FedTrace*, a generated-content-based watermark verification framework that attributes leaked federated models from suspicious generated images alone. FedTrace combines three components: a round-wise watermark lifecycle that separates identity distribution from utility aggregation, a reliability-aware carrier that concentrates identity evidence on low-drift watermark positions, and anti-collision coding with soft subset verification for collusive leakage. These components enable attribution after local training from generated outputs alone, without suspect weights or online query interfaces. Extensive experiments across customized diffusion datasets, base models, and federated settings show that FedTrace preserves detectable client identities after local training and strengthens subset-aware tracing against collusion.

1 Introduction

Federated learning (FL) enables collaborative model training without centralizing raw user data [23, 13]. Clients optimize models locally [15] within repeated distribution and aggregation cycles [3], with heterogeneous local objectives requiring careful optimization [17, 14]. Recent work addresses complementary challenges of domain-aware aggregation [34] and knowledge transfer across heterogeneous client models [18]. As large generative models become widely deployed and customized, this paradigm is increasingly attractive for adapting diffusion models and lightweight adapters under private client data. The same distribution step, however, creates a sharper theft surface than in many conventional FL applications: each participating client repeatedly receives an up-to-date model or adapter that can be copied, reused offline, and redistributed without the server seeing the misuse [29, 31]. For generative adapters, the stolen object can immediately produce marketable

*Corresponding author: Debin Liu (debinliuhust@gmail.com).

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

images, imitate protected styles, or supply downstream content markets. In such cases, the most visible evidence of misuse may be the generated content rather than the leaked checkpoint itself.

This setting exposes a gap in current federated model protection. A first class of methods follows a white-box verification paradigm: the defender embeds ownership or client-specific signals into model parameters, gradients, or internal representations, and later verifies a suspect model by inspecting its weights or extracted fingerprints [16, 22, 31]. Such methods can provide strong evidence when the leaked checkpoint is available, but this access is rarely guaranteed once an adapter is copied and redistributed. A second class follows a black-box paradigm: the defender prepares trigger sets, challenge inputs, or secret prompts, queries the suspect model, and infers ownership or client identity from its responses [6, 38, 41]. This removes the need for weight access, but still assumes an online model interface that the verifier can repeatedly probe. Both assumptions can be too strong for generative-model theft: a leaked adapter may be used privately, hidden behind another service, or used to generate images that are subsequently shared or sold, leaving the defender with neither suspect weights nor a queryable model, but only suspicious outputs already in the wild.

Recent diffusion-LoRA watermarking methods provide an important clue: customized generative adapters can carry identities that are recoverable from generated images [8, 30]. Yet these methods are primarily designed for centralized customization, where a trusted trainer can jointly optimize stylization and watermark embedding. This trusted-trainer assumption does not directly carry over to federated learning, where clients themselves may leak the distributed adapters. The watermark identities and the mechanisms for embedding and verifying them should therefore remain under server control and hidden from clients. Local optimization can weaken the identity embedded at dispatch, whereas aggregation can mix client-specific signals into the shared adapter. The challenge is therefore to preserve identities in client models without allowing them to interfere with shared-model learning. Thus, simply attaching a centralized diffusion watermark to each client does not solve the federated traitor-tracing problem.

We therefore study *generated-content-based traitor tracing in federated learning*: attributing a leaked client model from suspicious generated images alone. This formulation introduces three coupled challenges. First, identity insertion must fit a federated workflow in which clients perform ordinary local adaptation without access to the identity codebook or embedding rule. Second, the identity must survive after local training, where LoRA directions drift unevenly and some watermark bits become unreliable. Third, malicious clients may collude by merging or averaging their leaked adapters, causing overlapping identities to blur into ambiguous evidence. A merged adapter may retain evidence from several clients without closely matching any single identity. Identifying one likely contributor is therefore insufficient to recover the full set of clients involved.

We address these challenges with *FedTrace*, a generated-content-based watermark verification framework for federated learning. FedTrace is organized around three design choices. First, a round-wise watermark lifecycle injects client identities into dispatched adapters while stripping identity residuals before aggregation, so utility can accumulate globally without permanently storing client-specific marks. Second, a reliability-aware low-drift carrier concentrates identity evidence on watermark positions estimated to remain stable after local training. Third, anti-collision coding reduces overlap between client signatures, while soft subset verification compares the recovered evidence against candidate client combinations. In our diffusion-LoRA instantiation, the server performs watermark reinforcement while clients keep ordinary local image training, and verification requires neither suspect model weights nor query access.

Our contributions can be summarized as follows:

- To the best of our knowledge, this is the first work to formulate generated-content-based watermark verification for traitor tracing in federated learning, where leaked client models are attributed from outputs rather than from suspect weights or black-box queries.
- FedTrace, a diffusion-LoRA framework that couples dispatch-time identity imprinting, aggregation-time identity stripping, and next-round identity re-imprinting while leaving client-side learning as ordinary local image optimization.
- Reliability-aware low-drift embedding and verification, linking LoRA drift after local training to rank instability and bit unreliability for more robust generated-content attribution.
- Anti-collision identity design for federated tracing, keeping overlapping client signatures distinguishable under collusive model merging and subset-level attribution.

2 Related Work

Generative-model watermarking. Generative-model watermarking can be distinguished by whether the identity is attached after generation or embedded into the generation process itself. Output-side marks can be bypassed when the marking stage is omitted. Model-internal methods instead modify the sampling noise, as in Tree-Ring [36], or fine-tune the latent decoder, as in Stable Signature [9], so that generated images carry a recoverable signature. Other approaches authenticate generators through re-generation [7]. Attribution from generated images also includes origin attribution [35] and the analysis of generative-model fingerprints [40]. Within customized diffusion, AquaLoRA writes watermark information into the LoRA bottleneck through rank-wise modulation [8], while AuthenLoRA further entangles stylization and watermark decoding through a learned encoder–decoder pipeline [30]. These works show that customized diffusion adapters can carry recoverable identities, but they still assume a single-model training regime optimized for one owner. FedTrace follows the same protection spirit, but redesigns the construction for the federated learning pipeline, where identity preservation must coexist with repeated local adaptation and aggregation.

Federated model protection and traitor tracing. Continual FL methods study knowledge preservation under heterogeneous data [20] and resource constraints [19], while information-theoretic defenses address multimodal gradient leakage [37]. These efforts concern learning stability or data confidentiality; model protection instead concerns ownership and accountability after a model is distributed. Federated model protection studies how ownership signals, fingerprints, or tracing codes can survive collaborative training. Early methods such as WAFFLE and FedIPR focus on ownership verification under decentralized optimization without centralizing private data [32, 16]. Later work moves toward stronger client-level accountability: FedTracker combines ownership verification with client tracing [29]; FedFP studies anti-collusion fingerprint design [22]; FedRPW emphasizes privacy-compliant watermark splitting and staged verification [31]; RobWE extends this direction to personalized federated learning [39]; and FLClear explores visually verifiable multi-client watermarks [10]. TraMark [38], BlackCATT [25], and EmbTracker [41] further study traceable black-box verification. Related work investigates Tardos-based tracing codes [26] and the role of federated training dynamics in black- and white-box tracing [27]. Together, these studies address post-training ownership protection and traitor attribution in FL. However, most of them are developed for earlier FL settings centered on relatively simple discriminative models, rather than today's widely used generative-model pipelines.

Verification under limited model access. An equally important distinction lies in how much access the verifier has to the suspect model. White-box methods inspect signatures embedded in model parameters [33, 4] or internal activation distributions [28]; federated ownership and fingerprinting schemes likewise verify parameter-level evidence [16, 22]. Black-box methods instead query a suspect model using backdoor triggers [1, 5] or adversarial fingerprints [21]. Federated variants also rely on challenge-response access to verify ownership or trace clients [6, 25, 41]. In realistic model-theft scenarios, however, even query access may disappear after redistribution. A leaked adapter can be used privately, hidden behind another service, or converted into generated content sold downstream. In that regime, the defender may observe neither suspect weights nor an online API, but only images shared or published by others. FedTrace is designed for this stricter verification regime: it attributes leakage from suspicious generated images alone, making it closer to output-based forensic attribution than to standard white-box inspection or query-based black-box verification.

3 Preliminaries and Problem Statement

3.1 Federated learning paradigm

We consider a standard client–server FL system with client set $\mathcal{C} = \{1, \dots, N\}$. The server maintains a shared LoRA adapter Δ, selects a participating subset $\mathcal{S}^{(t)}$ at round t, and dispatches a trainable client model to each selected client. Client i optimizes a local objective on its private dataset $\mathcal{D}_i$ and returns an updated adapter after local training. A standard FL view is

$$\min_{\Delta} \frac{1}{N} \sum_{i=1}^{N} F_i(\Delta; \mathcal{D}_i), \qquad F_i(\Delta; \mathcal{D}_i) = \frac{1}{|\mathcal{D}_i|} \sum_{(x,y)\in\mathcal{D}_i} \ell_i(\Delta; x, y). \tag{1}$$

The server then aggregates the returned local models. Under a FedAvg-style rule,

$$\Delta^{(t+1)} = \sum_{i \in \mathcal{S}^{(t)}} \alpha_i^{(t)} \widehat{\Delta}_i^{(t)}, \qquad \alpha_i^{(t)} = \frac{|\mathcal{D}_i|}{\sum_{j \in \mathcal{S}^{(t)}} |\mathcal{D}_j|}. \tag{2}$$

FedTrace follows this same round structure, but replaces the ordinary broadcast model with a client-specific dispatched adapter $\widetilde{\Delta}_i^{(t)}$ whose identity is injected before local training and removed again before aggregation.

3.2 Threat model

Attack model. Following the commonly used threat model in federated watermarking and traitor tracing [29, 31], we treat every participating client as a potential model leaker. Clients are assumed to follow the prescribed local training protocol and return updates during normal FL execution, but they may also retain a copy of any dispatched client model. After receiving $\widetilde{\Delta}_i^{(t)}$, a malicious client may illegally redistribute that model, continue adapting it offline, or monetize it by generating and selling synthetic content. We consider both single-client leakage and collusive leakage [25, 38]. In the collusive case, a malicious subset $\mathcal{S}^\star \subseteq \mathcal{C}$ may share dispatched checkpoints, merge locally trained variants, or otherwise combine client-specific models before release.

Defense model. We build on the standard FedAvg-style client–server paradigm [23, 13], where the server dispatches a shared model, receives local updates, and aggregates them without accessing private client data. The server is trusted and acts as the defender, owner, and verifier of the FL system. It observes the shared model state and the returned local updates, keeps the watermark codebook and embedding procedure hidden from clients, and strips identity components before aggregation. At verification time, we assume the server does not have white-box access to suspect weights and does not rely on an online query API. Instead, it only receives a set of suspicious generated outputs

$$\mathcal{X}^\star = \{x_m^\star\}_{m=1}^M, \tag{3}$$

where M is the number of suspicious images and $x_m^\star$ denotes the m-th image. The server must determine whether these outputs originate from a leaked federated client model and, if so, which client or client subset is responsible.

3.3 Problem statement

The goal is to design a federated watermarking and verification mechanism that attributes leaked client models from suspicious generated outputs alone. At round t, the server dispatches a client-specific model $\widetilde{\Delta}_i^{(t)}$ to client i. After misuse occurs, the defender observes only $\mathcal{X}^\star$ and must infer the responsible source, either as a single client identity or, in the collusive case, as a client subset:

$$\widehat{\mathcal{S}} = f(\mathcal{X}^\star), \qquad \mathcal{S}^\star \subseteq \{1, \ldots, N\}. \tag{4}$$

When $|\mathcal{S}^\star| = 1$, the task reduces to standard single-traitor attribution; when $|\mathcal{S}^\star| > 1$, the task becomes collusion-aware subset attribution.

This problem imposes three requirements simultaneously. First, each dispatched client model should carry a client identity that remains decodable from generated outputs. Second, this identity should survive realistic local adaptation drift as much as possible. Third, client identities should remain distinguishable even when malicious clients merge or overlap their leaked models. FedTrace is designed to satisfy these three requirements in one round-wise verification framework.

4 Method

FedTrace is built around three coupled design choices, each addressing a failure mode in federated traitor tracing. The first is a *round-wise watermark lifecycle*: client identities must be present in dispatched models, but they should not permanently pollute the shared global adapter after aggregation. The second is a *reliability-aware identity carrier*: once clients continue local training, some LoRA directions drift much faster than others, so the watermark should concentrate on the most stable bit positions rather than treating all message bits equally. The third is *collusion-aware verification*: even

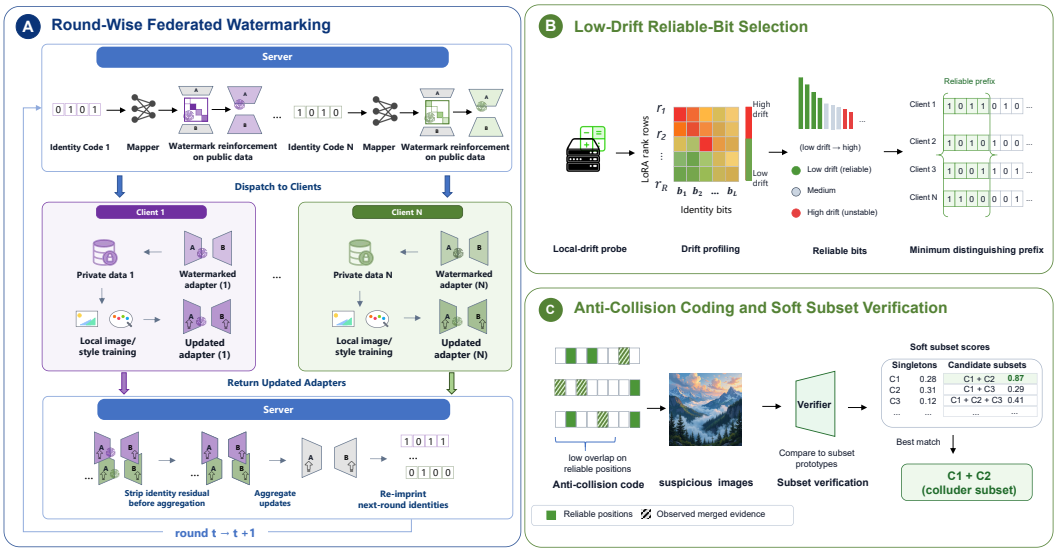

Figure 1: **Overview of FedTrace.** FedTrace protects federated diffusion-LoRA training through three coupled modules. (A) The server imprints client-specific identities into dispatched adapters, clients perform ordinary local image training, and the server strips identity-aligned residuals before aggregation and next-round re-imprinting. (B) A low-drift reliable-bit carrier selects watermark positions that remain stable after local training while still distinguishing active clients. (C) Anti-collision coding and soft subset verification attribute suspicious generated images to singleton clients or colluding client subsets without requiring suspect weights or an online query interface.

if single-client identities are separable, merged client models may still become ambiguous unless the code assignment and the detector are both designed with overlap in mind. Figure 1 summarizes the workflow, and we describe these three components in that order.

4.1 Round-Wise Federated Watermarking

The first challenge is to make client identity available at dispatch time without letting that identity accumulate inside the shared federated model. FedTrace therefore treats watermarking as a server-side, round-wise operation rather than a permanent modification of the global adapter. It starts from a watermark codec pretrained on a public dataset $\mathcal{D}_{\text{pub}}$ that is disjoint from all client-private data. The codec, consisting of a frozen secret encoder ψ_{enc} and decoder ψ_{dec}, is used only by the server during reinforcement and verification. At round t, the server assigns client i a binary code $c_i^{(t)} \in \{0,1\}^L$, maps it to a rank-wise modulation vector, and bakes the resulting identity into the LoRA down path:

$$s_i^{(t)} = \phi(c_i^{(t)}), \qquad \widetilde{W}_{\ell,\downarrow,i}^{(t)} = \text{Diag}(s_i^{(t)}) W_{\ell,\downarrow}^{(t)}. \tag{5}$$

Here ϕ is a server-side code-to-rank mapper with bit-embedding matrix $E \in \mathbb{R}^{L\times r}$, where r is the LoRA rank and ℓ indexes adapter layers. The down projection maps features into the rank dimension, and the up projection maps them back to the original feature dimension. Each row of E specifies how one identity bit contributes to the rank-wise modulation. The server bakes this modulation into the down-projection weights, leaving the up projection unchanged during imprinting; clients receive the resulting adapter, not a separate mapper. The server then runs a short public-data reinforcement stage so that the dispatched adapter remains both useful for generation and decodable by the frozen codec.

After local training, the returned adapter $\widehat{\Delta}_i^{(t)}$ contains both utility updates and the client-specific dispatch residual $R_i^{(t)} = \widetilde{\Delta}_i^{(t)} - \Delta^{(t)}$. Because identity is meant to be a distribution-time property, not something that should accumulate in the shared global state, FedTrace removes the component aligned with this residual before aggregation:

$$\overline{\Delta}_i^{(t)} = \widehat{\Delta}_i^{(t)} - \frac{\langle \widehat{\Delta}_i^{(t)} - \Delta^{(t)}, R_i^{(t)} \rangle}{\|R_i^{(t)}\|_2^2 + \varepsilon} R_i^{(t)}. \tag{6}$$

The stripped updates are then aggregated to obtain $\Delta^{(t+1)}$, and the next round re-imprints fresh client identities on top of that shared adapter. In this sense, FedTrace decouples utility accumulation from identity distribution: the shared adapter evolves by ordinary FL, while traceable identities are injected only when models are dispatched to clients. This strip–aggregate–re-imprint cycle is the basic systems-level scaffold that the later low-drift and collusion-aware components build on. Here and below, ε denotes a small positive constant used only for numerical stability.

4.2 Low-Drift Reliable-Bit Design

Once the server can reliably dispatch and strip identities, the next question is where those identities should be placed. The main difficulty is not dispatch-time decoding but post-dispatch drift: during local adaptation, some LoRA directions move much more than others, and bits coupled to those directions become much less trustworthy. FedTrace therefore estimates which bits are fragile and which are reliable, and then keeps only the shortest reliable prefix that still separates the active clients. The server profiles adaptation drift using proxy image training, without requesting private client data or client-side watermark computations. Let $d_{\ell,k}$ denote the relative Euclidean change of down-projection row k in layer ℓ, measured against its pre-adaptation norm. We aggregate these changes across layers, weighted by the corresponding pre-adaptation row norms, to obtain $\bar{d}_k^{(t)}$. The entry $E_{b,k}$ of the mapper's bit-embedding matrix measures the coupling between bit b and rank component k. We define bit unreliability and its reliability weight by

$$u_b^{(t)} = \frac{\sum_k |E_{b,k}|\, \bar{d}_k^{(t)}}{\sum_k |E_{b,k}| + \varepsilon}, \qquad w_b^{(t)} = (u_b^{(t)} + \varepsilon)^{-1}. \tag{7}$$

Here $u_b^{(t)}$ measures how strongly bit b is exposed to local drift, while $w_b^{(t)}$ is its reliability weight. Starting from a provisional assignment of distinct client codes, FedTrace sorts bits by increasing $u_b^{(t)}$ and retains only the shortest prefix of that ordering that already distinguishes the active clients; we denote this prefix by $\mathcal{R}^{(t)}$. Here, prefix means the first positions in the reliability ordering, not consecutive positions in the original code. Subsequent anti-collision code assignment preserves distinct client signatures on this prefix.

4.3 Anti-Collision Coding and Soft Subset Verification

Even after low-drift bit selection, single-client separability alone is not enough. If multiple clients share too many active positions on the retained prefix, their merged signatures can still collapse toward similar explanations. FedTrace therefore adds an anti-collision bias at the code-assignment stage. On the reliable prefix $\mathcal{R}^{(t)}$, client codes are not assigned arbitrarily; among codebooks that already distinguish the active clients, FedTrace prefers balanced assignments with large weighted separation and low overlap on active positions. We summarize this anti-collision preference by

$$\mathcal{C}^{(t)} = \arg\max_{\mathcal{C}} \min_{i \neq j} \left[\sum_{b \in \mathcal{R}^{(t)}} w_b^{(t)}\, \mathbf{1}\Big(c_{i,b}^{(t)} \neq c_{j,b}^{(t)}\Big) - \lambda \sum_{b \in \mathcal{R}^{(t)}} \mathbf{1}\Big(c_{i,b}^{(t)} = 1 \wedge c_{j,b}^{(t)} = 1\Big) \right]. \tag{8}$$

The first term rewards separation on reliable bits, while the second penalizes shared active positions that would make collusive mixtures collapse toward the same explanation; λ controls this separation–overlap tradeoff.

Given $\mathcal{X}^\star = \{x_m^\star\}_{m=1}^M$, let $z_{m,b}^{(1)}$ and $z_{m,b}^{(0)}$ denote the decoder logits for bit b. Single-client attribution uses weighted inner products with signed client codewords. For collusion tracing, let $\mathfrak{S}$ contain candidate singleton and collusive hypotheses, and let $\mu_{\mathcal{S}}$ denote the reference evidence prototype for a candidate $\mathcal{S} \in \mathfrak{S}$. We compare prototypes using weighted cosine similarity:

$$a_b = \frac{1}{M} \sum_{m=1}^{M} \left(z_{m,b}^{(1)} - z_{m,b}^{(0)} \right), \qquad \widehat{\mathcal{S}} = \arg\max_{\mathcal{S} \in \mathfrak{S}} \frac{\langle a, \mu_{\mathcal{S}} \rangle_w}{\|a\|_w \, \|\mu_{\mathcal{S}}\|_w}. \tag{9}$$

Here $\langle u, v \rangle_w = \sum_{b \in \mathcal{R}^{(t)}} w_b^{(t)} u_b v_b$ and $\|u\|_w = \sqrt{\langle u, u \rangle_w}$. Singleton prototypes average decoded evidence from reference images generated by each client adapter. Averaged subset prototypes average these singleton prototypes, whereas empirical subset prototypes use reference images generated by

merged adapters to capture merging effects. Normalization removes magnitude bias. Section 5.3 evaluates both constructions. Algorithm 1 in the appendix summarizes the complete dispatch–strip–aggregate–verify procedure.

5 Experiments

5.1 Evaluation Setup

Training setup. We evaluate FedTrace in a diffusion-LoRA federated personalization workflow. The server maintains a shared LoRA adapter, dispatches client-specific watermarked adapters at each round, and aggregates identity-stripped updates after local training. Unless otherwise stated, we use 16-bit client identities and run 20 communication rounds. Each client holds 300 private images and performs one local epoch per round. For watermarked runs, the server performs watermark reinforcement before the initial dispatch, followed by a shorter reinforcement stage after each aggregation and before the next dispatch. The encoder–decoder watermark codec introduced in Section 4.1 is pretrained once on public data and then frozen. Clients receive only the dispatched adapter and perform ordinary local image adaptation, without access to the watermark codebook, codec, or embedding procedure.

Implementation details. We implement the adapters using LoCon with rank $r = 160$ and scaling parameter $\alpha = 80$ for the UNet and text-encoder adapter modules. The pretrained UNet, VAE, and text-encoder backbone parameters remain frozen, while local optimization updates the adapter parameters using the ordinary image/style adaptation objective. Watermark reinforcement and identity stripping are performed by the server; clients neither optimize a watermark loss nor receive the codec or codebook. Appendix B provides the mapper parameterization and parameter-key details.

Table 1: Domains and backbones.

Dataset	Base model	Task
Realistic Human	SD v1.5	portrait
Anime Character	AnyLoRA	character
PixelArt	AnyLoRA	pixel-art
Lolita	MajicMix v7	clothing

Benchmarks and metrics. Table 1 summarizes the benchmark, with representative watermarked generations shown in Figure 2. For the stylization datasets, we follow AuthenLoRA's dataset choice and use the same four released domains: realistic portraits, anime characters, pixel-art images, and Lolita-style clothing designs [30]. These domains cover both realistic and cartoonish generation styles, as well as subject-, style-, and clothing-oriented personalization tasks. We use AnyLoRA for Anime Character and PixelArt, MajicMix v7 for Lolita, and Stable Diffusion v1.5 for Realistic Human. We evaluate generation quality using CLIP score and watermark verification after local training using bit accuracy, image-level top-1 and top-2 attribution accuracy, attribution margin, and client-level attribution accuracy. Unless otherwise specified, verification uses reliability-weighted soft decoding.

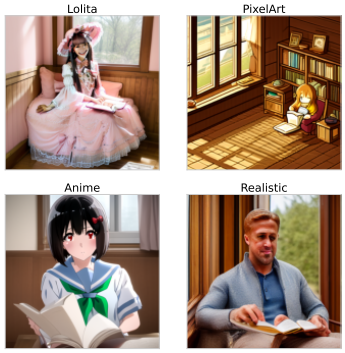

Figure 2: Watermarked generations across the four benchmark domains.

Adapted comparison methods. We adapt three representative protection strategies to verification from generated images, with watermark identities and embedding mechanisms kept under server control. *WAFFLE-style Global WM* adapts server-side FL watermark retraining by inserting one shared ownership watermark into the global adapter [32]; it can verify system ownership but does not assign client identities. *FedTracker-style Static FP* adapts global-local fingerprinting by assigning each client a fixed fingerprint at dispatch time, without round-wise stripping, refreshing, or low-drift code design [29]. *FedRight-style BB-FP* adapts black-box fingerprinting by constructing client-specific CLIP-space prototypes from images generated using reference prompts [6].

5.2 Main Results

Table 2 first compares FedTrace with adapted baselines on the representative 5-client realistic-human setting. All compared methods keep watermark identities and embedding mechanisms under server control, and verification uses generated images only. Clean FL gives the utility reference but has

Table 2: Comparison with adapted watermarking and fingerprinting baselines.

Method	Verification signal	Identity level	CLIP↑	Ownership↑	Client Acc.↑
Clean FL	none	none	33.68	–	–
WAFFLE-style Global WM	shared ownership watermark	system	32.95	0.92	–
FedTracker-style Static FP	fixed client fingerprint	client	32.74	0.88	0.40
FedRight-style BB-FP	generated-content prototype	proxy client	33.68	–	0.20
FedTrace	reliability-aware generated watermark evidence	client	32.98	0.94	0.80

Table 3: Watermark recovery and client attribution across four generation domains after local training.

Dataset	Base model	Verifier	BitAcc↑	Top-1↑	Top-2↑	Margin↑	Client Acc.↑	CLIP↑
Anime Character	AnyLoRA	Hard	0.872	0.612	0.901	0.421	0.600	31.64
		Soft	0.872	0.681	0.936	0.713	0.800	
		FedTrace	0.872	**0.764**	**0.963**	**1.284**	**0.800**	
PixelArt	AnyLoRA	Hard	0.906	0.684	0.948	0.782	0.600	33.78
		Soft	0.906	0.742	0.984	1.216	0.800	
		FedTrace	0.906	**0.823**	**0.993**	**2.031**	**0.800**	
Lolita	MajicMix v7	Hard	0.958	0.925	0.986	2.917	0.800	30.92
		Soft	0.958	0.962	0.992	3.684	1.000	
		FedTrace	0.958	**0.979**	**0.994**	**4.770**	**1.000**	
Realistic Human	SD v1.5	Hard	0.834	0.548	0.846	0.236	0.400	32.98
		Soft	0.834	0.603	0.884	0.489	0.600	
		FedTrace	0.834	**0.742**	**0.948**	**1.214**	**0.800**	

no traceability. WAFFLE-style global watermarking preserves an ownership signal, but its shared identity cannot distinguish clients. FedTracker-style static fingerprinting provides client-specific evidence after dispatch, yet degrades after local training because the fingerprint is never refreshed or protected from aggregation drift. FedRight-style generated-image prototypes require no explicit decoder, but their decisions are dominated by content and style similarity rather than stable client identity. FedTrace achieves stronger client attribution after local training by preserving reliable identity evidence, while retaining verification from generated images alone.

Table 3 reports watermark recovery and client attribution after local training across the four domains, using five clients and 495 evaluation images generated per client. We compare three verifiers on the same generated images: hard-bit matching thresholds each decoded bit before attribution, while soft-logit matching uses decoder confidence without reliability-based bit selection or weighting. BitAcc is averaged over decoded watermark bits. Top-1 and Top-2 measure whether the true client is ranked first or within the top two candidates for each generated image; Margin measures the score difference between the true client and the highest-scoring incorrect candidate; and Client Acc. measures attribution accuracy across clients, with each decision based on aggregated watermark evidence from all evaluation images generated by that client. This layout shows the central trend: most domains preserve a usable watermark signal after local training, while reliability-aware soft verification converts that signal into stronger client attribution.

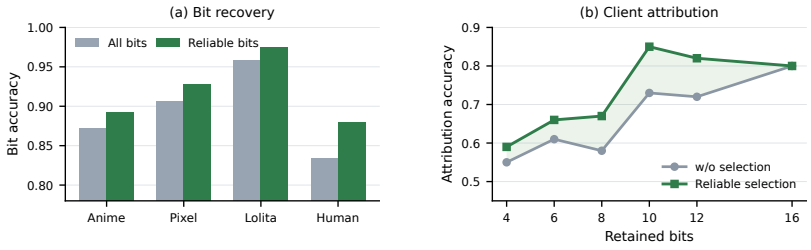

Figure 3: Effect of reliable-bit selection after local training. (a) Bit recovery across four domains using all positions or the selected reliable positions. (b) Client attribution as the retained prefix length varies, with and without reliability-based selection.

Reliable-bit ablation. Figure 3 separates watermark recovery from identity attribution. Evaluating only the selected reliable positions raises mean bit accuracy from 0.893 over all 16 positions to 0.919, with consistent gains across domains. Attribution fluctuates under short prefixes because too few positions may not separate all client codes. Once the prefix becomes sufficiently distinctive, reliability ranking reaches 0.85 at 10 bits, compared with 0.73 without selection. Adding higher-drift positions then reduces the advantage, and both variants reach 0.80 when all 16 positions are retained.

Generated-content attribution. Across datasets, generated images retain enough client-specific evidence for reliable attribution after local training. The trend is consistent across the four domains: hard-bit matching is already above chance, soft-logit matching improves image-level attribution by preserving decoder confidence, and the full FedTrace verifier further improves separation by emphasizing low-drift reliable positions. The gains are largest on the more visually entangled domains, where several clients share similar style prompts and a hard decoded codeword alone is more ambiguous. Figure 4 visualizes client-to-client matching scores in a representative five-client setting. Higher diagonal than off-diagonal scores indicate stronger evidence for the correct identities; negative scores indicate evidence against a candidate identity.

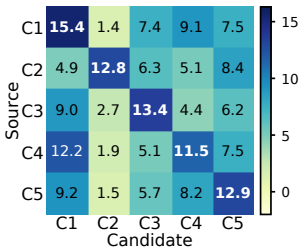

Figure 4: Representative client matching.

Client-scale effect. Moving from three to five clients generally reduces attribution accuracy, but the degradation is moderate when anti-collision coding and reliability-weighted verification are used. On the representative realistic-human setting, attribution decreases from 0.917 to 0.875 and 0.800 as the client count grows from 3 to 4 and 5, while CLIP stays in the same range ($33.04/32.89/32.98$). Thus, scaling mainly makes identity separation harder rather than reducing generation quality; the full table is in Appendix Table 6.

5.3 Collusion and Robustness

We next evaluate stronger leakage scenarios after local training. For collusion, we merge leaked client adapters from malicious two-client subsets and ask whether suspicious images are better explained by a singleton client or by a colluding subset. Collusion tracing identifies the contributing clients rather than only detecting system ownership. Table 4 reports all-pair two-client collusion over the 5-client setting. Singleton-only soft matching achieves an average any-colluder accuracy of 0.742 but does not identify colluding subsets. Any-colluder measures whether the prediction contains at least one true colluder; Subset-pref. measures whether a subset hypothesis beats all singleton identities; Exact subset requires recovering both colluding clients. Averaged subset prototypes are obtained by averaging the decoded singleton reference prototypes of the clients in each candidate subset. Empirical subset prototypes are instead constructed by merging the corresponding reference adapters, generating reference images, and averaging their decoded watermark evidence. Empirical subset prototypes improve exact subset recovery and separation from singleton hypotheses across all four domains, highlighting the benefit of constructing reference signatures from merged adapters.

Table 4: Collusion tracing across four generation domains after local training.

Dataset	Verifier	Any-colluder↑	Subset-pref.↑	Exact subset↑	Subset margin↑
Anime Character	Averaged subset prototype	0.808	0.641	0.294	0.149
	Empirical subset prototype	**0.887**	**0.932**	**0.601**	**0.377**
PixelArt	Averaged subset prototype	0.827	0.694	0.321	0.173
	Empirical subset prototype	**0.924**	**0.947**	**0.641**	**0.409**
Lolita	Averaged subset prototype	0.881	0.762	0.398	0.247
	Empirical subset prototype	**0.959**	**0.986**	**0.748**	**0.512**
Realistic Human	Averaged subset prototype	0.812	0.688	0.324	0.160
	Empirical subset prototype	**0.936**	**0.965**	**0.667**	**0.406**
Average	Averaged subset prototype	0.832	0.696	0.334	0.182
	Empirical subset prototype	**0.927**	**0.958**	**0.664**	**0.426**

For robustness, we apply both image-level post-processing and model-level transformations. Table 5 shows representative attacks: attribution is 0.736 under JPEG compression and 0.731 under Gaussian noise. Rank pruning reduces bit recovery to 0.761 and causes the largest attribution drop among the evaluated transformations.

Table 5: Robustness summary.

Level	Transformation	BitAcc	Top-1
Image	None	0.893	0.827
Image	JPEG Q=50	0.871	0.736
Image	Gaussian noise	0.810	0.731
Model	8-bit quant.	0.853	0.790
Model	Rank pruning	0.761	0.685

6 Conclusion

We presented FedTrace, a watermarking framework for tracing leaked federated client adapters from generated images without access to suspect model weights or a query interface. FedTrace integrates round-wise identity management, low-drift reliable-bit design, and anti-collision coding with subset-aware verification to support client attribution throughout federated training. Together, these components preserve client-specific watermark evidence through local adaptation and enable verification against both individual clients and colluding subsets. Experiments across four diffusion-LoRA domains demonstrate the benefits of reliability-aware decoding and empirical subset prototypes, while evaluating generation quality and robustness to image- and model-level transformations.

Acknowledgments and Disclosure of Funding

This work was supported by the National Key Research and Development Program of China under Grant 2024YFC3307900; the National Natural Science Foundation of China under Grants 62376103, 62302184, 62436003, 62472332, and 62672205; the Major Science and Technology Project of Hubei Province under Grants 2025BAB011, 2024BAA008, and 2024BAB039; the Hubei Science and Technology Talent Service Project under Grant 2024DJC078; the Science and Technology Innovation Talent Program under grant 2023CB005; and Wuhan Dameng Database Co., Ltd.

References

[1] Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *27th USENIX security symposium (USENIX Security 18)*, pages 1615–1631, 2018.

[2] Muhammad Ammad-Ud-Din, Elena Ivannikova, Suleiman A Khan, Were Oyomno, Qiang Fu, Kuan Eeik Tan, and Adrian Flanagan. Federated collaborative filtering for privacy-preserving personalized recommendation system. *arXiv preprint arXiv:1901.09888*, 2019.

[3] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečnỳ, Stefano Mazzocchi, Brendan McMahan, et al. Towards federated learning at scale: System design. *Proceedings of machine learning and systems*, 1:374–388, 2019.

[4] Huili Chen, Bita Darvish Rohani, and Farinaz Koushanfar. Deepmarks: A digital fingerprinting framework for deep neural networks. *arXiv preprint arXiv:1804.03648*, 2018.

[5] Huili Chen, Bita Darvish Rouhani, and Farinaz Koushanfar. Blackmarks: Blackbox multibit watermarking for deep neural networks. *arXiv preprint arXiv:1904.00344*, 2019.

[6] Jinyin Chen, Mingjun Li, Yao Cheng, and Haibin Zheng. Fedright: An effective model copyright protection for federated learning. *Computers & Security*, 135:103504, 2023.

[7] Aditya Desu, Xuanli He, Qiongkai Xu, and Wei Lu. Generative models are self-watermarked: Declaring model authentication through re-generation. *arXiv preprint arXiv:2402.16889*, 2024.

[8] Weitao Feng, Wenbo Zhou, Jiyan He, Jie Zhang, Tianyi Wei, Guanlin Li, Tianwei Zhang, Weiming Zhang, and Nenghai Yu. Aqualora: Toward white-box protection for customized stable diffusion models via watermark lora. In *International Conference on Machine Learning*, pages 13423–13444. PMLR, 2024.

[9] Pierre Fernandez, Guillaume Couairon, Hervé Jégou, Matthijs Douze, and Teddy Furon. The stable signature: Rooting watermarks in latent diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22466–22477, 2023.

[10] Chen Gu, Yingying Sun, Yifan She, and Donghui Hu. Flclear: Visually verifiable multi-client watermarking for federated learning. *arXiv preprint arXiv:2511.12663*, 2025.

[11] Pengfei Guo, Puyang Wang, Jinyuan Zhou, Shanshan Jiang, and Vishal M Patel. Multi-institutional collaborations for improving deep learning-based magnetic resonance image reconstruction using federated learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2423–2432, 2021.

[12] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

[13] Peter Kairouz and H Brendan McMahan. Advances and open problems in federated learning. *Foundations and trends in machine learning*, 14(1-2):1–210, 2021.

[14] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*, pages 5132–5143. PMLR, 2020.

[15] Jakub Konečnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.

[16] Bowen Li, Lixin Fan, Hanlin Gu, Jie Li, and Qiang Yang. Fedipr: Ownership verification for federated deep neural network models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4521–4536, 2022.

[17] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

[18] Yichen Li, Xiuying Wang, Wenchao Xu, Haozhao Wang, Yining Qi, Jiahua Dong, and Ruixuan Li. Feature distillation is the better choice for model-heterogeneous federated learning. *Advances in Neural Information Processing Systems*, 38:104726–104744, 2026.

[19] Yichen Li, Yuying Wang, Jiahua Dong, Haozhao Wang, Yining Qi, Rui Zhang, and Ruixuan Li. Resource-constrained federated continual learning: What does matter? *Advances in Neural Information Processing Systems*, 38:79493–79518, 2026.

[20] Yichen Li, Yuying Wang, Haozhao Wang, Yining Qi, Tianzhe Xiao, and Ruixuan Li. Fedssi: Rehearsal-free continual federated learning with synergistic synaptic intelligence. In *International Conference on Machine Learning*, pages 36151–36169. PMLR, 2025.

[21] Nils Lukas, Yuxuan Zhang, and Florian Kerschbaum. Deep neural network fingerprinting by conferrable adversarial examples. *arXiv preprint arXiv:1912.00888*, 2019.

[22] Yuling Luo, Yuanze Li, Sheng Qin, Qiang Fu, and Junxiu Liu. Copyright protection framework for federated learning models against collusion attacks. *Information Sciences*, 680:121161, 2024.

[23] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. Pmlr, 2017.

[24] Nicola Rieke, Jonny Hancox, Wenqi Li, Fausto Milletari, Holger R Roth, Shadi Albarqouni, Spyridon Bakas, Mathieu N Galtier, Bennett A Landman, Klaus Maier-Hein, et al. The future of digital health with federated learning. *NPJ digital medicine*, 3(1):119, 2020.

[25] Elena Rodríguez-Lois, Fabio Brau, Maura Pintor, Battista Biggio, and Fernando Pérez-González. Blackcatt: Black-box collusion aware traitor tracing in federated learning. *arXiv preprint arXiv:2602.12138*, 2026.

[26] Elena Rodríguez-Lois and Fernando Pérez-González. Towards traitor tracing in black-and-white-box dnn watermarking with tardos-based codes. In *2023 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–6. IEEE, 2023.

[27] Elena Rodríguez-Lois and Fernando Pérez-González. Exploring federated learning dynamics for black-and-white-box dnn traitor tracing. In *2024 2nd International Conference on Federated Learning Technologies and Applications (FLTA)*, pages 282–289. IEEE, 2024.

[28] Bita Darvish Rouhani, Huili Chen, and Farinaz Koushanfar. Deepsigns: A generic watermarking framework for ip protection of deep learning models. *arXiv preprint arXiv:1804.00750*, 2018.

[29] Shuo Shao, Wenyuan Yang, Hanlin Gu, Zhan Qin, Lixin Fan, Qiang Yang, and Kui Ren. Fedtracker: Furnishing ownership verification and traceability for federated learning model. *IEEE Transactions on Dependable and Secure Computing*, 22(1):114–131, 2024.

[30] Fangming Shi, Li Li, Kejiang Chen, Guorui Feng, and Xinpeng Zhang. Authenlora: Entangling stylization with imperceptible watermarks for copyright-secure lora adapters. *arXiv preprint arXiv:2511.21216*, 2025.

[31] Lujia Shi, Youliang Tian, Kunlong Jin, Shuai Wang, Changgen Peng, and Zhou Zhou. Fedrpw: Robust and privacy-compliant watermarking framework for federated model ownership protection. *Information Sciences*, page 122636, 2025.

[32] Buse GA Tekgul, Yuxi Xia, Samuel Marchal, and N Asokan. Waffle: Watermarking in federated learning. In *2021 40th International Symposium on Reliable Distributed Systems (SRDS)*, pages 310–320. IEEE, 2021.

[33] Yusuke Uchida, Yuki Nagai, Shigeyuki Sakazawa, and Shin'ichi Satoh. Embedding watermarks into deep neural networks. In *Proceedings of the 2017 ACM on international conference on multimedia retrieval*, pages 269–277, 2017.

[34] Haozhao Wang, Yichen Li, Wenchao Xu, Ruixuan Li, Yufeng Zhan, and Zhigang Zeng. Dafkd: Domain-aware federated knowledge distillation. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20412–20421. IEEE, 2023.

[35] Zhenting Wang, Chen Chen, Yi Zeng, Lingjuan Lyu, and Shiqing Ma. Where did i come from? origin attribution of ai-generated images. *Advances in neural information processing systems*, 36:74478–74500, 2023.

[36] Yuxin Wen, John Kirchenbauer, Jonas Geiping, and Tom Goldstein. Tree-ring watermarks: Fingerprints for diffusion images that are invisible and robust. *arXiv preprint arXiv:2305.20030*, 2023.

[37] Tianzhe Xiao, Yichen Li, Yining Qi, Yi Liu, Haozhao Wang, Yi Wang, and Ruixuan Li. Enhancing privacy in multimodal federated learning with information theory. *Advances in Neural Information Processing Systems*, 38:142199–142215, 2026.

[38] Jiahao Xu, Rui Hu, Olivera Kotevska, and Zikai Zhang. Traceable black-box watermarks for federated learning. *arXiv preprint arXiv:2505.13651*, 2025.

[39] Yang Xu, Yunlin Tan, Cheng Zhang, Kai Chi, Peng Sun, Wenyuan Yang, Ju Ren, Hongbo Jiang, and Yaoxue Zhang. Robwe: Robust watermark embedding for personalized federated learning model ownership protection. *arXiv preprint arXiv:2402.19054*, 2024.

[40] Ning Yu, Larry S Davis, and Mario Fritz. Attributing fake images to gans: Learning and analyzing gan fingerprints. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7556–7566, 2019.

[41] Haodong Zhao, Jinming Hu, Yijie Bai, Tian Dong, Wei Du, Zhuosheng Zhang, Yanjiao Chen, Haojin Zhu, and Gongshen Liu. Embtracker: Traceable black-box watermarking for federated language models. *arXiv preprint arXiv:2603.12089*, 2026.

A Algorithm and Round-Wise Dynamics

Algorithm 1 FedTrace: Roundwise Watermarking, Aggregation, and Verification

Require: public dataset $\mathcal{D}_{\mathrm{pub}}$, shared adapter $\Delta^{(t)}$, active clients $\mathcal{S}^{(t)}$, codec $(\psi_{\mathrm{enc}}, \psi_{\mathrm{dec}})$, mapper ϕ, aggregation operator $\mathcal{A}$

pretrain watermark codec $(\psi_{\mathrm{enc}}, \psi_{\mathrm{dec}})$ on $\mathcal{D}_{\mathrm{pub}}$ and freeze it
for each communication round t **do**
  at the server, profile proxy adaptation drift and compute bit unreliability using the mapper embeddings
  initialize distinct client codes, choose their minimum distinguishing reliable prefix $\mathcal{R}^{(t)}$, and refine the codebook $\mathcal{C}^{(t)}$ for anti-collision separation while preserving distinct prefix signatures
  for each client $i \in \mathcal{S}^{(t)}$ **do**
    compute rank modulation $s_i^{(t)} \leftarrow \phi(c_i^{(t)})$, imprint it into the LoRA down path, run short server-side reinforcement, and dispatch $\widetilde{\Delta}_i^{(t)}$
  end for
  for each client $i \in \mathcal{S}^{(t)}$ in parallel **do**
    run local image training and return the updated adapter $\widehat{\Delta}_i^{(t)}$ to the server
  end for
  for each returned adapter $\widehat{\Delta}_i^{(t)}$ at the server **do**
    form $R_i^{(t)} = \widetilde{\Delta}_i^{(t)} - \Delta^{(t)}$ and strip the identity-aligned component using Eq. (6) to obtain $\overline{\Delta}_i^{(t)}$
  end for
  the server aggregates $\{\overline{\Delta}_i^{(t)}\}_{i \in \mathcal{S}^{(t)}}$ to update the shared adapter $\Delta^{(t+1)}$
end for

Verification from suspicious generated images $\mathcal{X}^\star$:
decode logits $\{z_m\}_{m=1}^M$ from suspicious images, aggregate soft evidence $a_b = \frac{1}{M}\sum_m (z_{m,b}^{(1)} - z_{m,b}^{(0)})$, and return the highest-scoring singleton or subset hypothesis $\widehat{\mathcal{S}}$

Table 6: Client-scale effect on the representative realistic-human setting after local training. Margin measures the score difference between the true client and the highest-scoring incorrect candidate.

Clients	BitAcc↑	Attr.↑	Margin↑	CLIP↑
3	0.904	0.917	1.842	33.04
4	0.873	0.875	1.376	32.89
5	0.834	0.800	1.214	32.98

B Implementation Details

LoRA parameterization and injection site. We implement the LoRA adapters using LoCon while keeping the pretrained Stable Diffusion backbone frozen. For a LoRA module in layer ℓ, the effective weight update is

$$\Delta W_\ell = \frac{\alpha}{r} W_{\ell,\uparrow} W_{\ell,\downarrow},$$

where $W_{\ell,\downarrow}$ maps into the rank dimension and $W_{\ell,\uparrow}$ maps back to the original feature dimension. In our implementation, the client identity is injected as a rank-wise diagonal modulation on the down projection,

$$\widetilde{W}_{\ell,\downarrow,i} = \mathrm{Diag}(s_i) W_{\ell,\downarrow}, \qquad s_i = \phi(c_i) = \mathbf{1} + L^{-1/2} E^\top c_i,$$

where $E \in \mathbb{R}^{L \times r}$ is the mapper's bit-embedding matrix, L is the code length, and r is the adapter rank. Each row of E specifies the contribution of one identity bit to the rank-wise modulation. The same matrix is used to translate rank drift into bit unreliability in Eq. (7). The mapper is applied by the server, and its modulation is baked into the dispatched adapter rather than supplied as a separate client-side component. The watermark codec and client codebook are held by the server; clients receive only the dispatched adapter and do not receive the codec, codebook, or verification rule.

Server-side stripping implementation. The server-side stripping implementation operates on UNet LoRA down-projection deltas whose keys match `lora_unet.*.lora_down.weight`. It removes the identity-aligned component from the returned updates before aggregation and re-imprints client identities at the next dispatch.

Verification protocol. The verifier operates only on generated images. Each suspicious image is decoded by the frozen public-data codec to obtain bit logits; hard-bit accuracy is reported for diagnostic purposes, while attribution uses codeword-level matching over the active client codebook. Soft matching uses decoder logit gaps rather than only thresholded bits, and reliability-aware variants restrict or reweight matching to low-drift bit positions estimated from proxy local perturbations. For collusive leakage, FedTrace compares the suspicious evidence against candidate client subsets rather than only singleton identities.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes].

   Justification: The abstract and introduction state that FedTrace targets generated-content-based traitor tracing for federated diffusion-LoRA adapters, and the method and experiments are scoped to that setting.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [No].

   Justification: The paper states the threat-model assumptions and experimental scope but does not provide a dedicated discussion of limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A].

   Justification: The paper is primarily methodological and empirical, and does not claim formal theoretical theorems or proofs.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper?

   Answer: [Yes].

   Justification: The experimental protocol, datasets, base models, client counts, local training schedule, verification metrics, and implementation details are described in the main text and appendix.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No].

   Justification: The paper describes the implementation and uses publicly released datasets and base models, but an anonymized code release is not included in this manuscript package.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details necessary to understand the results?

   Answer: [Yes].

   Justification: Section 5 and the appendix specify the federated rounds, per-client data size, watermark reinforcement schedule, adapter parameterization, verification protocol, and attack settings.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No].

   Justification: The paper reports aggregate benchmark values and ablation trends without error bars or confidence intervals.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources needed to reproduce the experiments?

   Answer: [No].

   Justification: Training schedules are specified, but hardware specifications, runtimes, and total compute requirements are not fully reported.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics?

   Answer: [Yes].

   Justification: The work is a defensive watermarking and attribution study using released models and datasets, and does not involve human subjects, private personal data collection, or deployment.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No].

    Justification: The paper describes the defensive application but does not discuss both positive and negative societal impacts.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse?

    Answer: [N/A].

    Justification: The paper does not release a new pretrained image generator, dataset, or high-risk model asset; it studies a defensive attribution method.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets used in the paper properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [No].

    Justification: The paper cites the sources of the released domains, base-model choices, and related assets, but this manuscript does not yet enumerate explicit license names for every external asset.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A].

    Justification: The paper does not introduce or release a new dataset, benchmark asset, or pretrained model as a standalone artifact.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation?

    Answer: [N/A].

    Justification: The paper does not involve crowdsourcing experiments or human-subject studies.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed, and whether IRB approvals were obtained?

    Answer: [N/A].

    Justification: The work does not involve human subjects, so IRB approval is not applicable.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research?

    Answer: [N/A].

    Justification: LLMs are not used as a core methodological component of FedTrace.